Overview: Lab 1 develops the low level communications functions then lab 2 uses those to read and send voltage information from sensors. Lab 3 uses the functions from lab 2 along with interupt timers to create functions for PWM control. Lab 4 will likely use all the previous functions to read, evaluate, and consolidate additional sensors like encoders and IR sensors. Then finally Lab 5 will combine all of the functions together to create closed loop control for

	Function	Description	Location
Lab 1	void usb_read_next_byte()	Takes the next USB byte and reads it into a ring buffer for latter processing. If there is none waiting, it returns without blocking.	SerialO.h/c
	void usb_write_next_byte()	Takes the next byte from an output ring buffer and writes it to the usb port. If there is none waiting, it returns without blocking.	SerialO.h/c
	void usb_send_char(char)	takes a character and appends it to the output ring buffer	SerialO.h/c
	void usb_send_data(void*, uint8_t)	Takes a pointer to a buffer (everything can be void *) and the length of the buffer (hint use sizeof to help) and puts into the output ring	SerialO.h/c
	void usb_send_str(char*)	Puts a null-terminated c-string into the output ring buffer to send.	SerialO.h/c
	uint8_t usb_msg_length()	Returns the number of bytes in the receive buffer	SerialO.h/c
	uint8_t usb_msg_get()	Removes and Returns the next byte in the receive buffer	SerialO.h/c
	uint8_t usb_msg_peek()	Returns the next byte in the receive buffer but does not remove it.	SerialO.h/c
Summary	Using the USB_Echo_Task() as a template, earlier ring buffer code, and above functions to enable the car to act as a calculator. Need to note that USB interface only sends 8 Bits in a frame so don't block while sending larger messages (need to look at static CDC_LineEncoding_t LineEncoding1 in SerialO.c and understand what the function Endpoint_WaitUntilReady() is doing)		
Lab 2	read_voltage()	read voltage values from a sensor	
	send_voltage()	transmite voltage values to a sensor	
	signale_filter()	filter noisy sensor data	
Summary	Continually monitor the voltage readings from a sensor in order to use a physics model of the system to determine the best conversion of the direct voltage readings to the desired information about the system. In this case likely either vehicle speed or distance traveled.		
Lab 3	timer_counter_seq()	Set up timer counter that takes in the desired PWM, Timer/Counter Control Register value and other values to enable an enterupt	
	ISR using above timer	Actions to take when timer seq is completed and interupt occurs	
	pwm modes	Set up functions that run the motors and fast med and slow pwm options for vehicle control	
	invert pwm	set up function to allow changing motor directions	
Summary	Set up basic pwm control of system using timer interupts should be able to move the drive motors and different speeds and change direction of movement.		
Lab 4	read IR Sensors	use vehicles IR sensors for feedback on position?	
	read encoder sensors	use vehicles encoders for additional information regarding speed and movement	
Summary	Monitor the system as PWM signals are sent and drive directions given as well as the sensor readings to determine the dynamics of the overall system		
Lab 5	control function	given a set of drive directions the robot can execute them with low error	
Summary	Test system to determine accuracy and quality of control		